Approaching a Nanosecond : A 32 bit Adder

Gary Bewick Paul Song Giovanni De Micheli Michael J. Flynn

Computer Systems Laboratory
Stanford University

Abstract

This paper describes a high performance 32 bit binary adder designed at Stanford University, and fabricated at Signetics Inc. Measurements indicate that the adder computes the sum of two numbers (and a carry) in 2.1 nanoseconds and consumes 900 milliwatts using a power supply voltage of -4.5V. The adder is implemented using silicon emitter coupled logic (ECL) circuitry with 0.5 volt output swings. The high performance is a result of high speed logic/technology and a special addition algorithm which results in an adder with a maximum of 3 levels of logic from any input to any output. The maximum fanout on any signal is 8 input loads, the maximum number of inputs on any gate is 5, and the maximum number of WIRE-OR outputs is 8.

1 Introduction

An on-going project at Stanford is to investigate the fundamental operations performed in today's high performance digital systems, and to determine methods of implementing these operations as quickly as possible. Addition is one of these fundamental operations and has been used frequently to demonstrate fast technology [7] [2]. In this paper, a 32 bit binary ECL adder is presented, with an addition time of 1.8 nsec. The high performance of this adder is a result of the following :

- An addition algorithm that is a combination of the Ling adder [3], the carry lookahead adder [6], and the conditional sum adder [4]. This algorithm takes advantage of special circuits that can be built using ECL.
- Fast, silicon bipolar ECL implementation. Extensive use is made of the WIRE-OR capability of ECL.
- Increasing the driving current on critical high capacitance paths to reduce the delay.

The adder is a modular design, constructed from eight identical 4 bit slices, and a lookahead network which computes the carry to each slice.

2 The Addition Scheme

The addition process consists of three steps. Each of these operations requires time approximately equal to a single gate delay :

1. Each 4 bit slice produces carry generate/propagate information sent to the lookahead network. This process is similar to the scheme described by Ling, but only the WIRE-OR capability of ECL is needed, the WIRE-AND is not used.
2. The lookahead network computes the carry for each slice. Two copies of the group propagate signals come from each 4 bit slice to the lookahead network. One of these copies is used (in conjunction with a group propagate from an adjacent slice) to produce a propagate signal for an 8 bit group. This limits the fan-in requirements of the gates in the lookahead network to 5.
3. Each 4 bit slice uses the carry received from the lookahead network to compute the proper sum outputs.

The overall structure of the adder is shown in Figure 1.

The logical design of the adder assumes that all inputs and outputs, except the carry, are negative logic. It is a simple matter to prove that this is functionally indistinguishable from an adder with positive logic inputs and outputs, and negative logic carry.

2.1 Generate/Propagate

The gate level schematic for each 4 bit slice is shown in Figure 2. The operation of the carry generate/propagate information produced by each 4 bit slice can be understood by contrasting it with a conventional carry lookahead 4 bit slice. The conventional slice produces group carry generate (G) and group carry propagate (P) signals which are used as inputs to one or more levels of lookahead (Figure 3). The single group carry generate signal from the conventional slice is replaced by two signals, H and P_3. These two signals can be created with fewer gate delays than G. In addition to H and P_3, two copies of group P connect each 4 bit slice with the carry lookahead network, making a total of 4 signals that run from each 4 bit slice to the lookahead

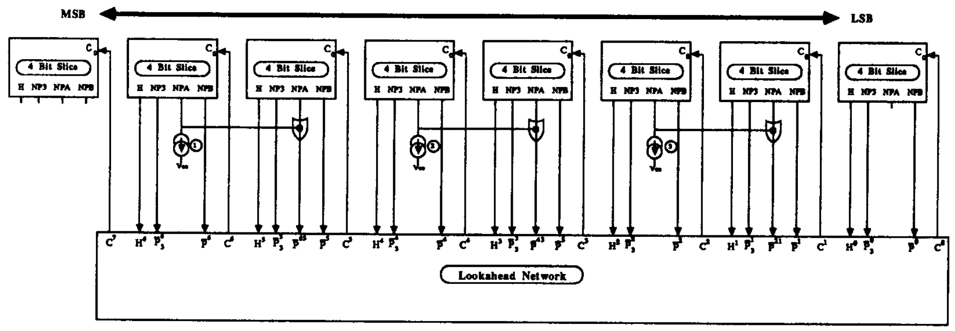

Figure 1: Adder Block Diagram

network. P_3 and the two copies of group P are sent in negative logic form. The relationship between G, H and P_3 is easily derived. G in a conventional carry lookahead 4 bit slice is :

$$G = G_3 + P_3G_2 + P_3P_2G_1 + P_3P_2P_1G_0 \quad (1)$$

where :

$$\begin{aligned} G_i &= A_iB_i \text{ (AND)} \\ P_i &= A_i \oplus B_i \text{ (EXCLUSIVE − OR)} \\ &\mathit{OR} \\ P_i &= A_i + B_i \text{ (OR)} \end{aligned}$$

A_i and B_i are the ith bits of the inputs to be to added (A_0 = LSB). Notice that the designer has a choice as to how to implement the P_i. They can be implemented as either an EXCLUSIVE-OR or an OR, or as some mixture of the two. If P_3 is chosen to be $A_3 + B_3$, then :

$$G_3 = P_3G_3$$

By using this for G_3, (1) becomes :

$$\begin{aligned} G &= P_3G_3 + P_3G_2 + P_3P_2G_1 + P_3P_2P_1G_0 \\ &= P_3(G_3 + G_2 + P_2G_1 + P_2P_1G_0) \\ &= P_3H \end{aligned}$$

where the new signal, H is defined as :

$$\begin{aligned} H &= G_3 + G_2 + P_2G_1 + P_2P_1G_0 \\ &= A_3B_3 + A_2B_2 + A_2A_1B_1 + B_2A_1B_1 \\ &\quad + A_2A_1A_0B_0 + A_2B_1A_0B_0 + B_2A_1A_0B_0 \\ &\quad + B_2B_1A_0B_0 \end{aligned}$$

Signal H is handy because it can be produced from the A and B inputs in a single gate delay, using gates with up to 4 inputs and WIRE-OR capability of 8. In comparison, G requires a minimum of 2 gate delays, one to compute the G_i and P_i required, and another to compute G. Alternately G could be expanded in terms of A_i and B_i, but this would require 15 WIRE-ORs and gates with up to 5 inputs. Referring to Figure 2, H is produced by the NOR gates numbered 1-8 (recall that the inputs are negative logic).

Since P is sent to the lookahead network in complemented form, it is easily formed by the WIRE-OR of the $\overline{P}_i$ from each bit position :

$$\begin{aligned} \overline{P} &= \overline{P_3P_2P_1P_0} \\ &= \overline{P}_3 + \overline{P}_2 + \overline{P}_1 + \overline{P}_0 \end{aligned}$$

$\overline{P}_i$ is produced by a special gate (Bit PG, described in section 3), which produces both $\overline{G}_i$ and $\overline{P}_i$ for each bit position (gates 10-13 in Figure 2). Internal to each slice, the P_i are implemented as $A_i \oplus B_i$, since the EXCLUSIVE-OR of A_i and B_i are needed for the final sum anyway. The $\overline{P}_3$ signal sent to the lookahead is produced by gate 9 in Figure 2.

2.2 Lookahead Network

A single level of lookahead is used to compute the carries to each 4 bit slice (Figure 4). To reduce the fan-in requirements of the gates in the lookahead network, the 4 bit slices are grouped in pairs, and a negative logic group propagate across 2 slices is produced by the WIRE-OR of one copy of the negative logic group propagates of the individual slices. Each 4 bit slice provides two copies of the negative logic group propagate (NPA and NPB in Figure 2), one of which is used in this WIRE-OR. The lookahead network accepts the group H and P_3 signals from each slice and combines them into group G in the same level of logic that is used to produce the carries to each 4 bit slice. For example, the carry to the most significant slice, C^7 is :

$$\begin{aligned} C^7 &= G^6 + P^6G^5 + P^6P^5G^4 + P^6P^5P^4G^3 + P^6P^5P^4P^3G^2 \\ &\quad + P^6P^5P^4P^3P^2G^1 + P^6P^5P^4P^3P^2P^1G^0 \\ &\quad + P^6P^5P^4P^3P^2P^1P^0C \end{aligned}$$

substituting $P_3^iH^i$ for each of the G^i :

$$\begin{aligned} &= P_3^6H^6 + P^6P_3^5H^5 + (P^6P^5)P_3^4H^4 + (P^6P^5)P^4P_3^3H^3 \\ &\quad + (P^6P^5)(P^4P^3)P_3^2H^2 + (P^6P^5)(P^4P^3)P^2P_3^1H^1 \\ &\quad + (P^6P^5)(P^4P^3)(P^2P^1)P_3^0H^0 \end{aligned}$$

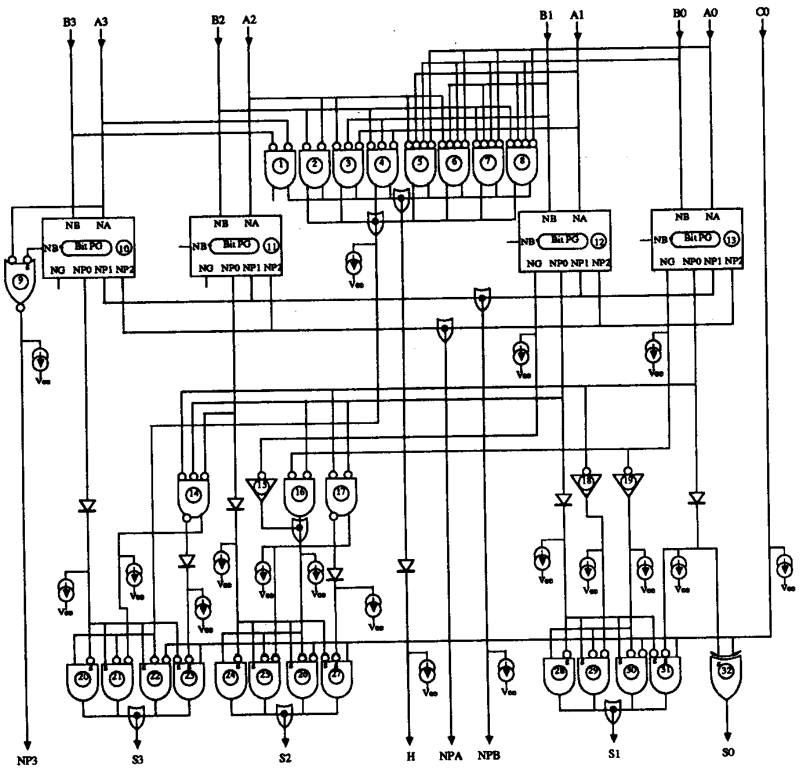

Figure 2: Four Bit Adder Slice

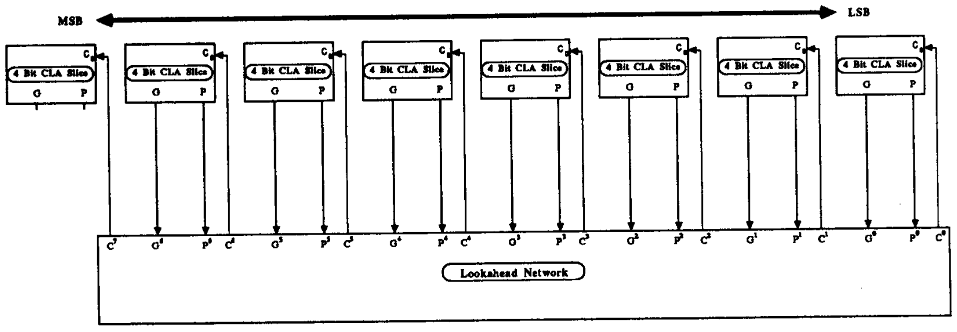

Figure 3: Conventional CLA Adder

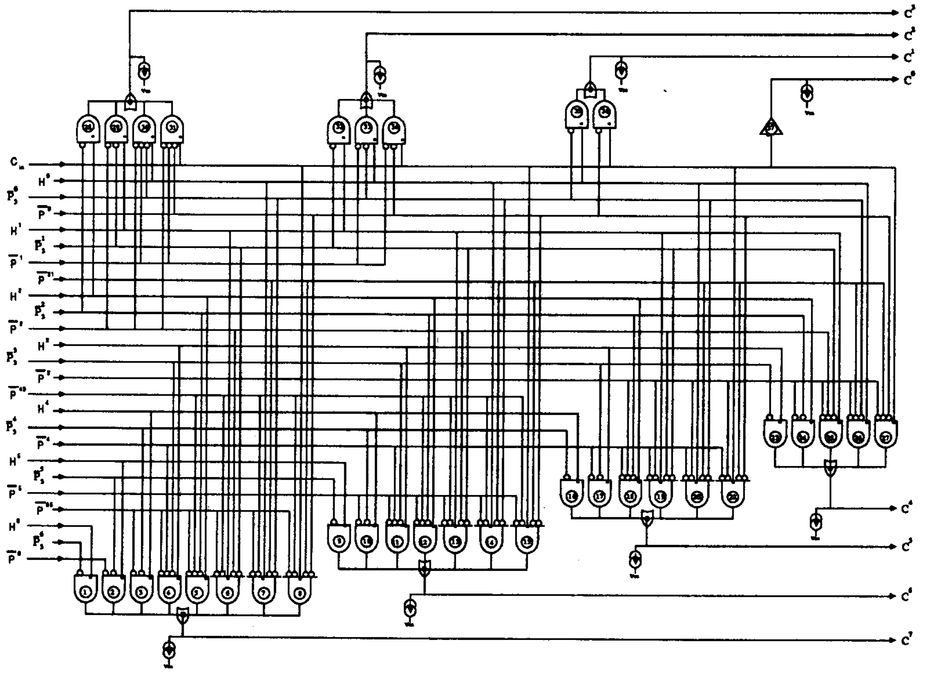

Figure 4: Lookahead Network

$$+ (P^6P^5)(P^4P^3)(P^2P^1)P^0C$$

$$\begin{aligned} = \; & P_3^6H^6 + P^6P_3^5H^5 + P^{65}P_3^4H^4 + P^{65}P^4P_3^3H^3 \\ & + P^{65}P^{43}P_3^2H^2 + P^{65}P^{43}P^2P_3^1H^1 \\ & + P^{65}P^{43}P^{21}P_3^0H^0 + P^{65}P^{43}P^{21}P^0C \end{aligned}$$

H^i, P^i, P_3^i and G^i are written to represent the H, P, P_3, and G signals from the ith 4 bit slice, respectively, (G is not actually produced by each slice). P^{ij} represents the propagate across the two adjacent slices i and j. Since all the P signals are negative logic, and the H signals are positive logic, a special kind of AND gate (described in section 3) with multiple inverting inputs and one non-inverting input is required. The carries to the other slices are produced in a similar manner.

2.3 Final Sum

At each 4 bit slice, special output logic is used to minimize the delay from the slice carry to the 4 slice outputs. The scheme used is related to conditional sum addition, but has less hardware overhead. Consider bit 3 (the MSB) of a 4 bit slice. The desired output, S_3, is the EXCLUSIVE-OR of the A_3 and B_3 inputs and the carry from the adjacent, less significant bit (bit 2 in this case) :

$$\begin{aligned} S_3 &= A_3 \oplus B_3 \oplus C_3 \\ &= E_3 \oplus C_3 \end{aligned} \tag{2}$$

This carry, C_3, is related to the slice carry, C, by the following equation :

$$\begin{aligned} C_3 &= G_2 + P_2G_1 + P_2P_1G_0 + P_2P_1P_0C \\ &= G_{2:0} + P_{2:0}C \end{aligned} \tag{3}$$

where :

$$\begin{aligned} G_{2:0} &= G_2 + P_2G_1 + P_2P_1G_0 \\ P_{2:0} &= P_2P_1P_0 \end{aligned}$$

That is, a carry reaches bit 3 if it is generated by one of bits 2,1, or 0 (all bits that are internal to the slice) and propagated to bit 3, or if the carry to the slice is propagated through bits 2,1, and 0 and the slice carry is a 1. (3) and (2) can be combined to give the output for S_3 :

$$\begin{aligned} S_3 &= E_3 \oplus (G_{2:0} + P_{2:0}C) \\ &= E_3\overline{G}_{2:0}\overline{P}_{2:0} + E_3\overline{G}_{2:0}\overline{C} + \overline{E}_3G_{2:0} + \overline{E}_3P_{2:0}C \end{aligned} \tag{4}$$

Since a negative logic output is needed, the complement of (4) is handy :

$$\overline{S}_3 = \overline{E}_3\overline{G}_{2:0}\overline{P}_{2:0} + \overline{E}_3\overline{G}_{2:0}\overline{C} + E_3G_{2:0} + E_3P_{2:0}C \tag{5}$$

There are similar equations for each of S_2, S_1, and S_0. (5) is implemented as the WIRE-OR of 4 gates for each output bit, except for S_0. In this case, (5) reduces to a simple EXCLUSIVE-OR of $\overline{E}_0$ and C. Referring to Figure 2, the gates numbered 20-23 are used to produce S_3. The process of selecting the proper sum bit with the slice carry is similar to the conditional sum or carry select adder in that the final sum is chosen by the slice carry (the second and fourth terms of (5)), but the selection process can be turned off if the final sum does not depend on the slice carry (the first and third terms).

3 The Circuits

The internal gates of the adder are constructed with bipolar ECL circuitry using 0.5 volt output swings. All outputs are buffered with emitter followers using constant current pulldowns. This allows the outputs of gates to be connected directly, with the resultant signal being a logic low only if all the outputs are low (WIRE-OR). All gates are run with tail currents of 500 μA, giving a basic gate delay of about 250 ps, not including any emitter follower delay. The emitter follower delay is dependent upon the capacitive load being driven. Most wires have a capacitance of the order of 100-200fF. These wires are run with emitter follower pulldown currents of 500 μA, giving a delay of 100 to 200ps. A number of wires (mostly those that connect the 4 bit slices to the lookahead network) have capacitances of up to 2.0pF. These wires are driven by emitter followers with currents of 2000 μA, giving delays of about 500 ps. The result is that the basic, loaded (NOR) gate has a delay of 350 to 750ps.

The adder implementation uses three major gate types :

- Conventional ECL NOR gates.
- A special AND gate with up to 4 inverting inputs and a single non-inverting input (Figure 5). The non-

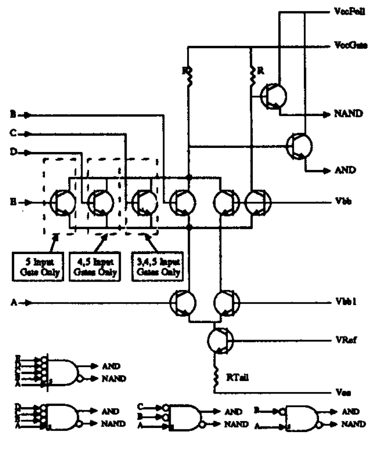

Figure 5: Special AND gate

inverting input must be level translated by a single diode drop to avoid saturating the input transistor. The delay from the translated input to either output is slower (about 100ps) than the untranslated inputs. There is also a version of this gate in which the inverting and non-inverting inputs are interchanged. The alternate version is used when the delay from the non-inverting input must be minimized (for example gates 23, 27 and 31 of Figure 2).

- A gate for generating the complements of the bit P_i and G_i with a single tail current, shown in Figure 6. As in the special AND gate, the B input is slower than the A input.

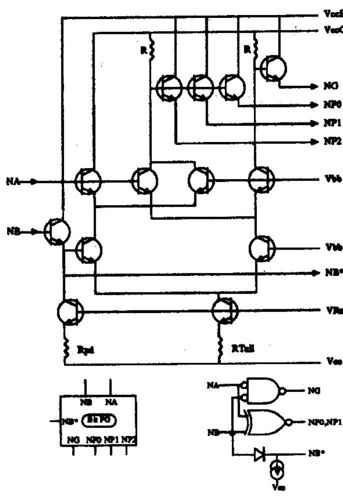

Figure 6: Bit PG Circuit

4 The Chip

The layout of the adder was done using a set of design rule independent module generators, written in the L language of the GDT 3.1 environment provided by Silicon Compiler Systems Inc. Two goals have been achieved by using design rule independent module generators :

- Use of an aggressive, industrial bipolar process in the frame of university/industry cooperation. Much of the layout was done without knowledge of the specific design rules.
- Achieve portability of the adder layout over future sets of design rules, anticipating use of the adder as a building block in larger functional units.

The adder generator is procedural and hierarchical. The 4 bit slice generator is invoked eight times and the lookahead generator is invoked once to assemble the chip layout. The 4 bit slice and the lookahead are implemented by two columns of gates and wired by a channel router. Gate generators assemble and connect the bipolar transistors, diodes and resistors according to the required gate

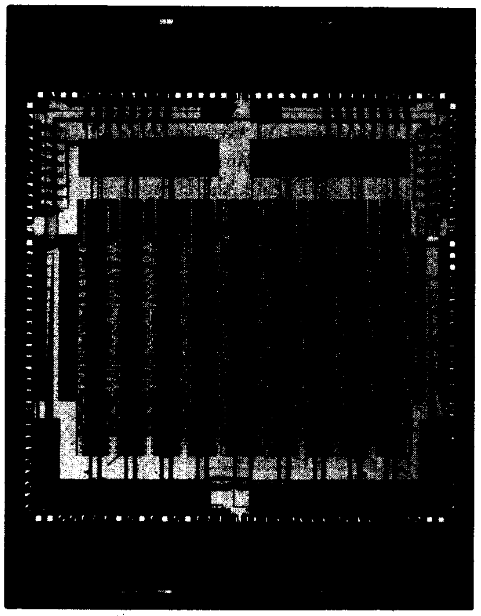

Figure 7: Adder die. Actual size : 6.6mm x 6.6mm

function and tail current level. The circuit was fabricated using HS3.5 bipolar technology from Signetics. Figure 7 is a photograph of the die. The area of just the adder portion is approximately 5.4mm x 4.6mm. Each dark vertical stripe represents either one 4 bit slice or the look-ahead network (the longer stripe in the center is the lookahead network). Inputs run from the bottom two thirds of the I/O pads into the logic, and outputs run from the logic to the top one third of the pads. Outputs (H, P_3, P) from the 4 bit slices to the lookahead network run above the stripes, while carries from the lookahead to the slices run below the stripes. Approximately 50% of the adder area is dedicated to power supply routing. Since a fairly small voltage swing is being used, the supply buses are made wide enough to restrict V_{cc} (logic swings are referenced to V_{cc}) differences across the die to less than 40mV.

Measurements of the adder indicate an addition time of 2.1 nsec, not including package or output driver delay. The chip consumes 200ma (again not including output driver current) using a power supply of -4.5 volts.

5 Conclusion

The design presented above does not perform addition in less than a nanosecond, but it does come close enough to indicate that such performance may be possible. Improvements can be made in both the speed and power consumption by using a more careful layout to reduce the length of the interconnections. The power consumption may also be minimized by reducing the value of current sources along non-critical paths. Another area of investigation is the extension of the design to word lengths of 54-64 bits for use with double precision floating point.

6 Acknowledgements

Special thanks to the people at Signetics for their assistance and support, in particular Peter Baltus, Ron Cline, and Uzi Bar-Gadda. In addition Mark Horowitz of Stanford provided invaluable advice and assistance. This project is funded by a seed grant from the Center for Integrated Systems (CIS) at Stanford University. The chip was fabricated at Signetics Inc., Sunnyvale California.

References

[1] M. I. Elmasry. *Digital Bipolar Integrated Circuits.* John Wiley & Sons, New York, New York, 1983.

[2] Inseok S. Hwang and Aaron L. Fisher. A 3.1ns 32b CMOS adder in multiple output domino logic. In *1988 IEEE International Solid-State Circuits Conference*, pages 140–141, 1988.

[3] H. Ling. High-speed binary adder. *IBM Journal of Research and Development*, 25(2 and 3):156–166, May 1981.

[4] J. Sklansky. Conditional sum addition logic. *Transactions of the IRE*, EC-9(2):226–230, June 1960.

[5] S. Waser and M. J. Flynn. *Introduction to Arithmetic for Digital Systems Designers.* Holt, Rinehart and Winston, 1982.

[6] A. Weinberger and J. L. Smith. A one-microsecond adder using one-megacycle circuitry. *IRE Transactions on Electronic Computers*, EC-5:65–73, June 1956.

[7] Ryuichiro Yamamoto, Asamitsu Higashisaka, Shuji Asai, Tautomu Tsuji, Yoichiro Takayama, and Seiken Yano. Design and fabrication of depletion GaAs LSI high-speed 32-bit adder. *IEEE Journal of Solid State Circuits*, SC-18(5):592–599, October 1983.